Nie mamy przed sobą jeszcze wizji zinformatyzowanej Polski, kraju, w którym każda pojedyncza inicjatywa, każda praca mogłaby być wspierana (mikro)komputerem, narzędziem efektywniejszym od poprzednich. To przyszłość trudna do wyobrażenia. Niemniej dostrzegamy już sporo przejawów zmiany. Zmiany sposobu myślenia o informatyce, o własnym w niej udziale. Informatyka przestaje być u nas rzeczywistością samoistną, zaczyna wreszcie zbliżać się do życia — indywidualnego, zbiorowego... Tak, konstatacja ta nie jest powszechna, a nasz KLAN ma w istocie wymiar mikro. Ale zaczyna być klanem ludzi rozumianych...

Pamięci kasetowe stają się przeżytkiem. W krajach o przeciętnie rozwiniętej technice komputerowej stosowane są już tylko w prymitywnym sprzęcie, spełniającym rolę „inteligentnej" zabawki.

Całkiem odmiennie wygląda ta sprawa u nas. Magnetofon kasetowy jest w zasadzie jedynym ratunkiem dla hobbystów, ba — często nawet dla profesjonalnych użytkowników mikrokomputerów. O pamięciach domenowych, EAROM-ach (ang. Electricaly Alterable Read Only Memory) trudno nawet marzyć. Nabycie stacji dysków elastycznych (nawet tej Made in Poland) przedstawia się nader mgliście. W ciągu ostatniego roku zetknąłem się tylko z jednym szczęśliwcem, któremu udało się tego dokonać. Ale skoro mógł on zainstalować komputer ODRA na ciężarówce i sterować czymś tam na poligonie, to i w potyczkach „o dyski" miał zapewne specjalny priorytet.

W mikrokomputerach dla zapisu informacji na taśmie magnetycznej wykorzystywane są zazwyczaj metody o „filozofii" zbliżonej do kodu Kansas City. W realizacji sprzętowej toru odczytu wykorzystuje się dwa filtry częstotliwościowe. Na wyjściu jednego pojawia się sygnał dla „1", na wyjściu drugiego dla „0".

Poniżej zamieszczamy opis rozwiązania, w którym odczyt i zapis informacji zrealizowano całkowicie programowo. Koszt konstrukcji podraża niestety konieczność zastosowania programowanego licznika (nie produkowany w kraju). Opisane rozwiązanie wykorzystane zostało w systemie MSA80.1000 opracowanym w Przemysłowym Instytucie Elektroniki w Warszawie i wdrażanym obecnie do produkcji w zakładach ZUMPE w Szczytnie. Ponieważ w tym systemie mikroprocesor równocześnie obsługuje klawiaturę i wyświetlacze cyfrowe, prędkość transmisji wynosi tylko 150 bodów. Jeżeli równolegle z obsługą magnetofonu mikroprocesor nie musi realizować innych zadań, można tę prędkość zwiększyć.

Programowa obsługa pamięci kasetowej

Podstawowym problemem przy przechowywaniu informacji cyfrowej na taśmie magnetycznej jest wybranie takiego sposobu kodowania, który pozwala osiągnąć możliwie dużą odporność na zakłócenia i przekłamania. Z reguły konieczny jest pewien kompromis między niezawodnością a szybkością przesyłanej informacji. W proponowanym rozwiązaniu stanowi logicznemu „0" odpowiada osiem impulsów o częstotliwości 1200 Hz, a stanowi logicznemu „1" — cztery impulsy o częstotliwości 600 Hz. Szybkość transmisji wynosi w tym przypadku 150 bodów. Transmisja jest asynchroniczna, z dwoma bitami stopu.

Organizacja logiczna przesyłanych zbiorów może być zrealizowana różnymi sposobami i nie będzie tu analizowana. Opisana zostanie natomiast realizacja zapisu i odczytu informacji na poziomie bitu.

W przedstawianym rozwiązaniu do współpracy z mikrokomputerem można wykorzystać magnetofon kasetowy MK

ZWIASTUN

Rozmowa z LESZKIEM WILKIEM, prezesem Klubu Mikrokomputerowego ABAKUS

μK: — Sytuacja kreuje Pana na przywódcę polskiej mikroinformatyki. Niemal zewsząd słyszymy Pana nazwisko...

Leszek Wilk: — Może dlatego, że jestem jednym z niewielu, którzy myślą o mikroinformatyce w kategoriach społecznych. Jakieś piętnaście lat temu zrozumiałem, że komputer zapowiada rewolucję. Przygotowywałem się. Czekałem, aż będzie w moim zasięgu. Dopiero w ciągu ostatnich dwóch lat ceny mikrokomputerów tak spadły, że mogłem za własne pieniądze kupić ZX81 — i wystartować.

Zainteresowanie mikroinformatyką w Polsce jest ogromne. Świadczą o tym rozmowy z ludźmi. I fakt, że coraz więcej osób posiada własny sprzęt. Ja sam zainteresowany jestem najbardziej wykorzystaniem komputerów do edukacji. Wprawdzie nie jestem informatykiem, ale miałem sprzęt i pomysły — poszukałem więc innych. Tak powstał prywatny klub, początkowo u mnie w domu.

Leszek Wilk (z lewej) w pojedynku z komputerem

μK: — A skąd pierwsi klubowicze?

L.W.: — Koledzy lub ludzie z ogłoszenia, które dałem do gazety. Ale zgłosili się głównie profesjonaliści. A z profesjonalistami pracuje się dość ciężko. Oni są w stanie zrobić poważne, rozbudowane programy — translatory, kompilatory, etc. Natomiast zastosowań, które bezpośrednio służyłyby człowiekowi, nie specjaliście, jakoś nie potrafią realizować. Nie mają tego w sobie.

Ale gdybym nie ja zaczął — byłby ktoś inny. To się musi lada chwila zacząć na dobre. Musi eksplodować. Pierwszy krok zrobią ogólnodostępne kluby. W dzisiejszej sytuacji — przy braku sprzętu i oprogramowania, ale przy sporym zainteresowaniu — to dla nas jedyna droga. A zaczynać trzeba szybko.

μK: — To jest poczucie obowiązku czy misji?

L.W.: — Jestem ekonomistą. Napisałem kiedyś pracę we Francji, po francusku. Omawia obecną sytuację ekonomiczną świata i na tym tle — ekonomię Polski, a w szczególności handlu zagranicznego. Wnioskiem z tej pracy może być stwierdzenie, że motorem rewolucji, jaka

Zwiastun

odbywa się obecnie w ekonomii światowej, jest mikroprocesor i że inwestycje powinny pójść już dawno w zupełnie innym kierunku, co oznacza — między innymi — ogromną szansę dla Polski. To zresztą jest jednocześnie szansa i przymus. Ekonomia światowa dematerializuje się, tzn. zaczyna opierać się na dwóch głównych filarach: informacji i edukacji. Mówi zresztą o tym ostatni Raport Rzymski. Z drugiej strony — do obecnego poziomu cywilizacyjnego człowiek po prostu nie dorasta. Nie dlatego, że nie potrafi — ma w końcu ogromne możliwości, w małym tylko stopniu wykorzystywane — ale dlatego, że nie jest odpowiednio wyedukowany. Trzeba więc szukać metod, abyśmy mogli dorosnąć do warunków, jakie zostały już stworzone, abyśmy mogli przeskoczyć do tej nowej, oczekującej nas — cywilizacji informacyjno-edukacyjnej. Byłem uczniem, studentem, a potem nauczycielem w szkole — i jestem przekonany, że jest to możliwe tylko z pomocą mikrokomputerów.

µK: — Ale Pan nie rozróżnia kryzysu ekonomii światowej od kryzysu u nas, jednak wyraźnie odmiennego.

L.W.: — Nieprawda. W gruncie rzeczy problemy są te same. Byłem we Francji, i znam jej gospodarkę. Mimo różnicy ustrojów, jest wiele identycznych problemów. Za dwa—trzy lata będą mieć te same trudności, które my mieliśmy w roku 1979.

µK: — Tam jednak trwa boom informatyczny, a u nas...

L.W.: — Oni też niewiele zrobili. Owszem, jest tam ileś set tysięcy mikrokomputerów. Ale jaki jest z nich pożytek?! To są w przewadze ogłupiające gry. W Europie dotychczasowe oprogramowanie „edukacyjne" to właściwie nic wobec ogromu potrzeb — jakieś pojedyncze, przypadkowe programy, zamiast wielotysięcznego rynku. Lepiej jest tylko w Stanach Zjednoczonych.

Ekonomia „materialna", rozwinięta na Zachodzie, która za swój główny cel ma wzrost produkcji, już się kończy. Tak jak kiedyś gwałtownie zmniejszyło się znaczenie rolnictwa na rzecz przemysłu, tak teraz przemysł ustąpi miejsca edukacji. Ale Zachód bardzo wolno zmienia kurs. Mamy więc szansę ich dogonić. Zamiast bawić się teraz w produkcję rzeczy materialnych, których zresztą nie umiemy produkować (z jednej tony surowca wytwarzamy znacznie mniej niż na Zachodzie), możemy od razu przeskoczyć do „niematerialnej". Mamy w niej większe szanse.

µK: — Zadziwiający jest ten Pana optymizm. Jak my, przy naszych zaściankowych umysłach, kiepskiej edukacji, z poczuciem bezradności w wielu sferach życia zbiorowego, mamy tego dokonać?

L.W.: — Najpierw trzeba przygotować ludzi, A ludzie nie są przystosowani do wymogów dnia dzisiejszego — tu, i tam. Przeciętnie to jest ten sam poziom. Tak więc barierę stanowi konieczność zdecydowanej inwestycji w edukację. My możemy to zrobić wcześniej. Umysły „zachodnie", opanowane zasadami starej ekonomii, nie są w stanie dokonać szybkiego przewrotu. Oni myślą kategoriami produkcji krótko- i średniookresowej, kryteriami zysku. My, nie mając nic do stracenia, możemy myśleć perspektywicznie. Teoretycznie biorąc, nam łatwiej byłoby zainwestować w masową edukację.

µK: — No dobrze, niebawem będzie miał Pan w Klubie tysiąc osób przy mikrokomputerze. I co dalej?

L.W.: — Nie! To nie jest tak. Przyszłość mikroinformatyki nie jest już dzisiaj zależna od sprzętu, lecz od oprogramowania. Wartość sprzętu stanowi jakieś 10—30% wartości całego komputera. A my możemy wyprodukować oprogramowanie...

µK: — Ale jak rozwijać oprogramowanie bez sprzętu? I to jeszcze masowo.

L.W.: — To prawda, nie można — w tej tzw. dużej informatyce, gdzie wymagany stopień komplikacji jest znacznie 232 (lub podobny). Magnetofon jest dołączony do mikrokomputera za pośrednictwem trzech linii sygnałowych — zapisu, odczytu i włączenia napędu. W celu zmniejszenia kosztów konstrukcji, wszystkie trzy linie obsługiwane są programowo. Układowo zrealizowano jedynie przetworzenie sygnałów o poziomach TTL na sygnały wchodzące lub wychodzące z magnetofonu. Dodatkowym elementem systemu mikroprocesorowego jest programowany licznik (np. 8253), generujący przerwanie po czasie zadanym przez procesor.

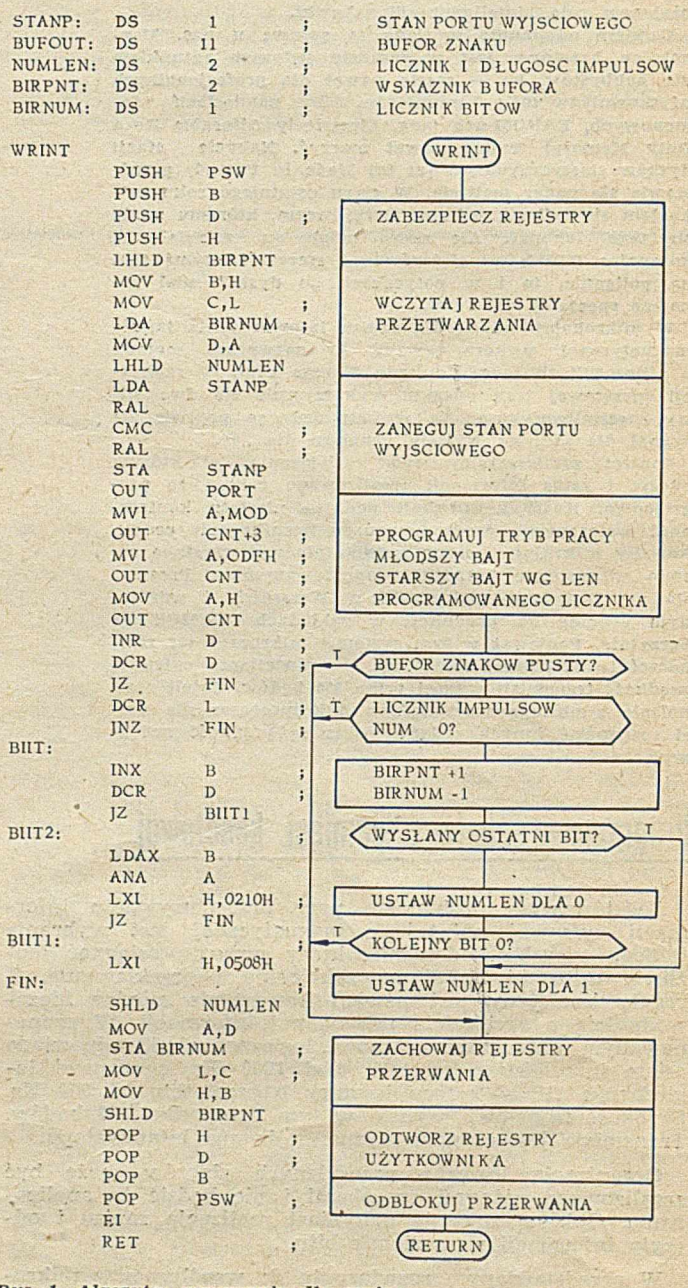
Zapis

Przy zapisie sygnału licznik generuje przerwanie co 835 µs dla bitu „1" lub co 417 µs w przypadku bitu „0". Program obsługi przerwania wykorzystuje 15 bajtów pamięci RAM, w których przechowuje swoje zmienne.

BUFOUT o długości 10 bajtów przeznaczony jest na przechowywanie aktualnie wyprowadzanego znaku. Każdemu bitowi znaku przyporządkowano jeden bajt w bufo-

```
STANP:  DS    1          ;   STAN PORTU WYJSCIOWEGO
BUFOUT: DS    11         ;   BUFOR ZNAKU
NUMLEN: DS    2          ;   LICZNIK I DLUGOSC IMPULSOW
BIRPNT: DS    2          ;   WSKAZNIK BUFORA
BIRNUM: DS    1          ;   LICZNIK BITOW

WRINT                    ;
        PUSH  PSW
        PUSH  B
        PUSH  D          ;   ZABEZPIECZ REJESTRY
        PUSH  H
        LHLD  BIRPNT
        MOV   B,H
        MOV   C,L        ;   WCZYTAJ REJESTRY
        LDA   BIRNUM     ;   PRZETWARZANIA
        MOV   D,A
        LHLD  NUMLEN
        LDA   STANP
        RAL
        CMC              ;   ZANEGUJ STAN PORTU
        RAL              ;   WYJSCIOWEGO
        STA   STANP
        OUT   PORT
        MVI   A,MOD
        OUT   CNT+3      ;   PROGRAMUJ TRYB PRACY
        MVI   A,ODFH     ;   MŁODSZY BAJT
        OUT   CNT        ;   STARSZY BAJT WG LEN
        MOV   A,H        ;   PROGRAMOWANEGO LICZNIKA
        OUT   CNT        ;
        INR   D          ;
        DCR   D          ;   T BUFOR ZNAKOW PUSTY?
        JZ    FIN
        DCR   L          ;   T LICZNIK IMPULSOW
        JNZ   FIN        ;     NUM  0?
BIIT:
        INX   B          ;   BIRPNT +1
        DCR   D          ;   BIRNUM -1
        JZ    BIIT1
BIIT2:                   ;   WYSŁANY OSTATNI BIT?  T
        LDAX  B
        ANA   A
        LXI   H,0210H    ;   USTAW NUMLEN DLA 0
        JZ    FIN
BIIT1:                   ;   T KOLEJNY BIT 0?
        LXI   H,0508H
FIN:                     ;   USTAW NUMLEN DLA 1
        SHLD  NUMLEN
        MOV   A,D
        STA BIRNUM       ;   ZACHOWAJ REJESTRY
        MOV   L,C        ;   PRZERWANIA
        MOV   H,B
        SHLD  BIRPNT
        POP   H          ;   ODTWORZ REJESTRY
        POP   D          ;   UŻYTKOWNIKA
        POP   B
        POP   PSW        ;   ODBLOKUJ PRZERWANIA
        EI
        RET              ;   RETURN
```



Rys. 1. Algorytm przerwania dla zapisu

rze, dzięki czemu znacznie skrócono czas obsługi przerwania. Pierwszy bajt bufora znaku równy jest 00H (bit startu), a dwa ostatnie mają wartość 0FFH (dwa bity stopu). W zmiennej BIRPNT przechowywany jest wskaźnik aktualnie wyprowadzonego bitu. Zmienna BIRNUM zawiera licznik bitów, które pozostały do wyprowadzenia. Jeżeli osiągnie on wartość 0, automatycznie wyprowadzane będą impulsy o częstotliwości 600 Hz. Zmienna NUM zawiera informację o tym, ile przerwań zostanie przyjętych dla wyprowadzenia całego bitu (8 dla bitu „1" i 16 dla bitu „0"). Zamienna LEN przechowuje wartość, przepisywaną do programowanego licznika, określającą czas trwania kolejnego impulsu.

Algorytm obsługi przerwania dla zapisu przedstawia rysunek 1.

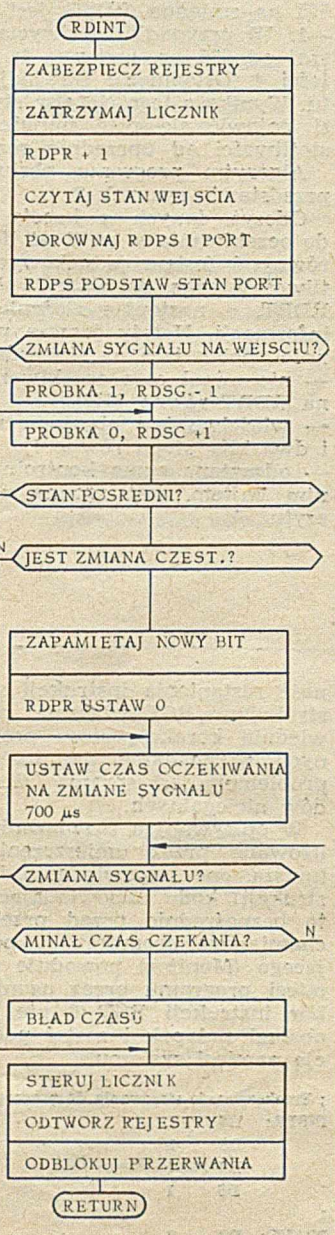
Dla użytkownika systemu zapis danych na taśmę magnetyczną sprowadza się do kolejnego wpisywania bajtów do bufora BUFOUT i oczekiwania w pętli, aż bajt zostanie

```
RDBT:   DS    1      ;   WARTOSC BITU
RDPR:   DS    1      ;   LICZNIK PROBEK
RDPS:   DS    1      ;   POCZATKOWY SYGNAL
RDSC:   DS    1      ;   LICZNIK SYGNALU

RDINT:                ;
        PUSH  PSW
        PUSH  H       ;
        MVI   A,MOD1
        OUT   CNT+3   ;
        LXI   H,RDPR  ;
        INR   M       ;
        IN    PORT
        CMA           ;
        ANI   80H
        INX   H       ;
        CMP   M
        MOV   M,A     ;

        INX   H       ;
        JZ    RDITO   ;
        DCR   M
        DCR   M       ;
RDITO:
        INR   M       ;
        MVI   A,0FFH
        JM    RDIT1
        MVI   A,04H   ;
        CMP   M       ;
        JNC   RDIT2
RDIT1:                ;
        MOV   M,A
        ANI   80H
        LXI   H,RDBT
        CMP   M       ;
        JZ    RDIT2
        MOV   M,A
        INX   H       ;
        MVI   M,00H
RDIT2:
        LDA   RDPS    ;
        MOV   L,A     ;
        MVI   H,30H   ;
RDIT3:
        IN    PORT
        ANI   80H     ;
        CMP   L
        JZ    RDIT4
        DCR   H       ;
        JNZ   RDIT3

        LXI   H,RDBT
        MVI   M,01H   ;
RDIT4:
        MVI   A,0
        OUT   CNT     ;
        MVI   A,4
        OUT   CNT     ;
        POP   H
        POP   PSW     ;
        EI
        RET           ;
```



Rys. 2. Algorytm programu obsługi przerwania dla odczytu

Zwiastun

większy. Ale nie w mikroinformatyce. Tu najważniejszy jest pomysł. Proszę przypomnieć sobie, jak rozwijała się mikroinformatyka na Zachodzie. To nie IBM, nie CDC ją rozwinęły, ale... dwuosobowa (na początku) APPLE. A w Europie? To przede wszystkim pojedynczy ludzie w Anglii, nie potentaci wielkiego przemysłu.

Jeśli natomiast w Polsce do końca tego roku powstanie dwadzieścia ogólnodostępnych klubów, takich jak mój, w największych miastach — to dla wszystkich twórczych ludzi starczy sprzętu, literatury i podstawowego oprogramowania.

µK: — Powiedzmy, że za rok będzie nawet więcej niż dwadzieścia otwartych klubów. I co dalej? W co może się — w wymiarze społecznym — przetworzyć ta praca kilkudziesięciu grup zapaleńców?

L.W.: — Powstaną dobre programy, głównie gry edukacyjne i programy do przetwarzania informacji (np. VisiCalc). W ciągu dziesięciu lat powinno ich powstać kilka, kilkanaście tysięcy.

µK: — I wykorzystywać je będą członkowie klubów...

L.W.: — Nie, one zostaną oddane Ministerstwu Oświaty, Ministerstwu Szkolnictwa Wyższego, przedszkolom, szkołom — wszystkim zainteresowanym. Będą sprzedawane wraz ze sprzętem, którego już wówczas będzie dużo. Jeśli mikrokomputer będzie kosztował — przy dzisiejszych cenach — 50—70 tys., to już będzie można mówić o dostępności. Nabytek taki może się nawet finansowo opłacać, gdy — powiedzmy — dzięki komputerowi będzie można nauczyć się jako tako języka obcego w kilkanaście tygodni.

µK: — Jaki ma być ten człowiek, którego chciałby Pan wyedukować?

L.W.: — Twórczy, myślący szerzej, szybko i efektywnie, nie jak dzisiaj. Dam przykład. Poszedłem do dyrekcji Domów Centrum z propozycją zorganizowania stoiska informacyjno-reklamowego związanego z usługami. Zostałem przyjęty na stojąco. Gdy powiedziałem o komputerach, ta pani mnie niemal wyrzuciła za drzwi. Gdybym był szewcem czy krawcem, to proszę — możemy rozmawiać. Oni chcą robić usługi i na nic innego nie mają czasu. Gdy właśnie głównym problemem usług jest niedostateczna informacja — kto, co i gdzie wykonuje, na jakich zasadach. Komputerowy system zbierający informacje o usługach Domów Centrum i wokół nich byłby niezwykle cenny.

W Krakowie zwróciłem się do Urzędu Miasta z propozycją uruchomienia produkcji mikrokomputerów. Odpowiedzieli, że teraz potrzebują odżywek dla dzieci i narzędzi chirurgicznych, komputerami nie są zainteresowani. Nie rozumieją, że z eksportu mikrokomputerów byłyby i odżywki, i te narzędzia. Oni patrzą krótko — odżywka to odżywka, nic dalej.

µK: — To jest Pana najsilniejszy przeciwnik — okrzepła biurokracja.

L.W.: — To nawet nie to. Ludzie myślą zbyt krótkowzrocznie. Dzisiaj głównym problemem ekonomicznym jest nie zadłużenie czy marnotrawstwo, ale poziom umysłowy ludzi.

µK: — I to będzie Panu przeszkadzało.

L.W.: — Ale nie przeszkodzi! Hamulce trzeba osłabiać.

µK: — To są silne hamulce. Wiele dążeń już zatrzymały.

L.W.: — Życie bez problemów niewiele byłoby warte. Nie wystarczy głośno krzyczeć, trzeba jeszcze umieć przekonać. Każdy problem jest do rozwiązania, trzeba tylko znaleźć sposób. Zresztą, z ekonomicznego punktu widzenia, nawet tego krótkookresowego — mam rację.

µK: — Gdybyż rozstrzygnięcia ekonomiczne decydowały!

L.W.: — Tu zadecydują. To jest lawina, która już się toczy. Owszem, gdybym podjął się zawracania kijem rze-

Zwiastun

ki, to pewnie byłyby kłopoty. To byłoby wbrew naturze, prawom fizyki. Ale to właśnie moi przeciwnicy próbują to czynić. Człowiek jest na mikroinformatykę skazany, dlatego też ani mnie, ani nikomu innemu — nie przeszkodzą.

μK: — Co więc z tym krajem? Jaki będzie po rewolucji głoszonej przez Pana?

L.W. — Szalone upowszechnienie mikrokomputerów, gwałtowny rozwój edukacji — to zupełnie zmieni sytuację, przeobrazi ten kraj, nas wszystkich. Być może ominiemy okres fascynacji głupimi grami komputerowymi, właśnie kończący się na Zachodzie, od razu realizując program masowej mikroinformatycznej edukacji. Będziemy tworzyć za pomocą szarych komórek — surowca, którego nigdy nam nie braknie. Wystarczy tylko je rozwijać.

μK: — Czy spotyka się Pan ze zrozumieniem ze strony osób, które w ogóle zechciały Pana wysłuchać?

L.W.: — Ludzi na ogół nie interesuje szersze uzasadnienie. Myślą, jak zdobyć ciekawą grę, jak ją potem dobrze sprzedać, jak wykorzystać mikrokomputer za darmo, albo jak go z korzyścią wyprodukować. Dominuje krótkowzroczność, dążenie do zysku — za tydzień, za dwa.

μK: — I nie ma nikogo, kto dotrzymywałby Panu kroku?

L.W.: — Prawdę mówiąc — nie spotkałem. Ale to nic, niebawem się pojawią.

Opracował:
ZBIGNIEW GLUZA
Fot. TOMASZ MALANOWSKI

pasjonaci wszystkich pokoleń

μK

skupiajcie się!

wyprowadzony przez autonomiczną procedurę. Przerwy między zbiorami generowane są automatycznie do czasu zablokowania przerwań.

Odczyt

Program analizujący sygnał odczytu wykorzystuje ten sam programowany licznik, generujący przerwanie po 625 μs od chwili zainicjowania. Program wykorzystuje 4 bajty pamięci RAM.

Zmienna RDBT zawiera ostatnio rozpoznany bit — i przyjmuje wartość 80H dla bitu „1", 00H dla bitu „0" i 01H, jeżeli minął czas oczekiwania na zmianę sygnału wejściowego (700 μs po przerwaniu). Zmienna RDPR zawiera licznik rozpoznanych impulsów sygnału wejściowego. Licznik jest zerowany po stwierdzeniu, że nastąpiła zmiana częstotliwości (pięć jednakowych impulsów). Zmienna RDPS zawiera stan linii wejściowej przy poprzednim przerwaniu. Umożliwia to stwierdzenie, czy nastąpiła zmiana sygnału.

Zmienna RDSG służy do określenia czy następuje zmiana częstotliwości. Przyjmuje ona wartość —1 dla częstotliwości 1200 Hz i wartość 4 dla częstotliwości 600 Hz. W przypadku wykrycia impulsu o długości mniejszej od 625 μs zmienna RDSG jest pomniejszana aż do wartości —1. W przypadku wykrycia impulsu o długości większej od 625 μs, zmienna ta jest powiększana aż do wartości 4. Osiągnięcie skrajnej wartości oznacza zmianę bitu. Wynika z tego, że aby stwierdzona została zmiana, musi pojawić się pięć impulsów więcej o odmiennej częstotliwości od uprzednio rozpoznanej.

Algorytm programu obsługi przerwania dla odczytu przedstawia rysunek 2.

Odczyt danych z kasety magnetycznej sprowadza się do oczekiwania w pętli aż licznik RDPR (modulo 8) będzie równy 0 (osiem jednakowych próbek lub zmiana częstotliwości) oraz do odczytu w tych momentach zmiennej RDBT, a następnie złożenia z odczytanych bitów pełnej informacji. Należy jeszcze przeprowadzić analizę poprawności, polegającą na stwierdzeniu, czy:
— nie minął czas oczekiwania na kolejną próbkę (zmienna RDBT równa 01H)
— właściwe jest obramowanie bajtu (bit startu równy 0 i dwa bity stopu równe 1)
— odczytana suma kontrolna, zapisana z reguły za ostatnim bajtem, jest zgodna z sumą obliczoną podczas odczytu.

ADAM PLUTA
Przemysłowy Instytut Elektroniki
Warszawa

Dobre narzędzia to połowa sukcesu. Dlatego też poświęcamy im wiele miejsca. Opisywaliśmy już sposób programowej realizacji pracy krokowej, niebawem zamieścimy rozwiązanie sprzętowe... Nie jest bowiem prawdą (ach ci dziennikarze-popularyzatorzy o bezkresnej ignorancji), że programiście wystarczy papier i dobrze naostrzony ołówek. Papier i ołówek może posłużyć do napisania artykułu do mikroKLANU, program zaś — trzeba uruchomić!

W „skrzynce narzędziowej" programisty (programy: Monitor, DDT — ang. Designer Development Tool) najczęściej znaleźć można realizację tzw. chwilowych puntków wstrzymania. Spełniają one pokładane w nich nadzieje, dopóki nie przyjdzie nam testować programów iteracyjnych. Ustawianie pułapki przy każdym obrocie pętli może wzbogacić o niejeden siwy włos. Stąd koncepcja stałych pułapek.

Stałe pułapki

Realizacja nie kasujących się automatycznie pułapek (punktów wstrzymania) wymaga wykonania śledzonej instrukcji poza uruchamianym programem (np. w obszarze RAM zarezerwowanym na potrzeby systemu operacyjnego). Jedną z pojawiających się trudności jest konieczność zastąpienia instrukcji wywołania podprogramu (z instrukcjami RST włącznie) przez instrukcje skoku i odpowiednią korekcję stosu programu z uwzględnieniem różnych przypadków dla instrukcji warunkowych. Drugim problemem jest sprawa interpretacji instrukcji HLT i kodów nielegalnych.

W opisywanym rozwiązaniu punkty wstrzymania są realizowane przez umieszczenie pod wskazanymi adresami (muszą one odpowiadać adresom pierwszych bajtów instrukcji) kodu odpowiadającego RST 0. Wykonywane jest to bezpośrednio przed przekazaniem sterowania do uruchamianego programu, a powrót do programu zarządzającego (Monitor) powoduje zawsze odtworzenie stanu pamięci programu przez usunięcie wszystkich pułapek. Wybór instrukcji RST 0 nie jest krytyczny dla algorytmu obsługi pułapek i został podyktowany przyjętą konfiguracją pamięci systemu.

```
; Przekazywanie sterowania do programu
INSTR:  DS   1                    ; miejsce wykonywania instrukcji
        DS   1
        DS   1
        DS   1
;                                 pseudorejesty
PSAVE:  DS   2                    ; PC
SSAVE:  DS   2                    ; SP
```

```
ASAVE:  DS    2                    ; AF
HSAVE:  DS    2                    ; HL
DSAVE:  DS    2                    ; DE
BSAVE:  DS    2                    ; BC
;
;       sekwencja startu
;
RUN:    LXI   H, (NOP)+(NOP) SHL 8
        SHLD  INSTR
        MVI   H, (JMP (PSAVE))
        SHLD  INSTR+2
        LHLD  PSAV                 ; pobranie kodu
        MOV   C, M
        XCHG                       ; DE := adres
        CALL  LENGTH
        MOV   C, A                 ; długość instrukcji
        LXI   H, INSTR
COPY:   DCR   C                    ; wpisanie do INSTR
        JM    TEST
        LDAX  D
        MOV   M, A
        INX   D
        INX   H
        JMP   COPY
TEST:   MOV   A, B                 ; badanie kodu instrukcji
        CPI   (CALL 0)             ; jeżeli CALL. zastąpienie
        JZ    REPL                 ; przez JMP i korekta stosu
        ANI   (RST 0)
        CPI   (RST 0)              ; jeżeli RST wydzielenie
        JNZ   Ccnd?                ; adresu i dalej jak wyżej
        XRA   B
        MOV   L, A
        MVI   H, 0
        SHLD  INSTR+1
        JMP   REPL
Ccnd?:  CPI   (CNZ 0)              ; jeżeli wywołanie warunkowe,
        JNZ   GO                   ; to test na efektywność.
        XRA   B                    ; wydzielenie kodu warunku
        RAR
        RAR
        RAR
        RAR
        MOV   C, A                 ; 0-Z, 1-C, 2-P, 3-S
        SBB   A
        MOV   B, A                 ; rodzaj testu, 0 dla false
        LDA   ASAVE
        ANI   0C5H                 ; wydzielenie bitów warunkow.
        MOV   L, A
        RLC
        SBB   A
        MVI   H, 10H
        ANA   H
        ORA   L
SEL:    RRC
        RRC
        DCR   C
        JP    SEL
        ANA   H
        XRA   B
        ANA   H
        JNZ   GO                   ; skok przy nieefektywnym
REPL:   LHLD  SSAVE
        DCX   H                    ; korekcja stosu programu
        MOV   M, D
        DCX   H
        MOV   M, E
        SHLD  SSAVE
        MVI   A, (JMP 0)           ; JMP zamiast Ccnd
        STA   INSTR
GO:     XCHG                       ; adres po instrukcji
        SHLD  PSAVE
        ;
        ; ustawienie pułapek
        ;
        LXI   SP, SSAVE            ; odtworzenie rejestrów
        POP   H
        POP   PSW
        POP   D
        POP   D
        POP   B
        SPHL
        LHLD  HSAVE
        JMP   INSTR                ; i wreszcie START!
```

Przedstawiony algorytm, realizujący przekazanie sterowania do uruchamianego programu, wykorzystuje fragment pamięci wydzielony dla Monitora.

```
; Procedura LENGTH
; DZIAŁANIE: podaje ilość bajtów instrukcji, kodom nielegalnym i instrukcji
; HLT przypisuje zerową długość.
; WEJŚCIE: rej. C — kod instrukcji,
;          TABLE — tablica długości.
; WYJŚCIE: rej. A — ilość bajtów,
;          rej. B — kod instrukcji.
; ZMIENIA: B, C, H, L, A, F.
;
LENGTH: MOV   B, C                 ; przechowanie kodu
        MOV   A, C
        SUI   (HLT)
        RZ                         ; powrót z 0 dla HLT
        ADI   (HLT)+(MOV B, B)
        MOV   C, A                 ; C := kod + 40H
        SUI   (MOV B, B)+(MOV B, B)
        SBB   A
        INR   A                    ; 1 dla kodów 40-BF
        RNZ
        ADD   C                    ; A := C, CY:= 0
        RAR
        ORA   A
        RAR                        ; numer wiersza
        MOV   C, A
        MOV   A, B                 ; kod instrukcji
        MVI   B, 0
        LXI   H, TABLE             ; tablica długości
        DAD   B
        MOV   B, A                 ; kod instrukcji
        ANI   3
        MOV   C, A                 ; numer kolumny
        MOV   A, M
LNO:    RLC                        ; długość
        RLC                        ; do b1, b0
        DCR   C
        JP    LNO
        ANI   3
        RET
; Tablica długości przypisywanych kodom z zakresów
; OOH-3FH i OCOH-OFFH, zorganizowana w 32 wiersze
; po 4 kolumny. Element tablicy zajmuje 2 bity
;
TABLE:  DB    01011111B    ; RNZ  POP   JNZ   JMP
        DB    11011001B    ; CNZ  PUSH  ADI   RST
        DB    01011100B    ; RZ   RET   JZ
        DB    11111001B    ; CZ   CALL  ACI   RST
        DB    01011110B    ; RNC  POP   JNC   OUT
        DB    11011001B    ; CNC  PUSH  SUI   RST
        DB    01001110B    ; RC         JC    IN
        DB    11001001B    ; CC         SBI   RST
        DB    01011101B    ; RPO  POP   JPO   XTHL
        DB    11011001B    ; CPO  PUSH  ANI   RST
        DB    01011101B    ; RPE  PCHL  JPE   XCHG
        DB    11001001B    ; CPE        XRI   RST
        DB    01011101B    ; RP   POP   JP    DI
        DB    11011001B    ; CP   PUSH  ORI   RST
        DB    01011101B    ; RM   SPHL  JM    EI
        DB    11001001B    ; CM         CPI   RST
        DB    01110101B    ; NOP  LXI   STAX  INX
        DB    01011001B    ; INR  DCR   MVI   RLC
        DB    00010101B    ;      DAD   LDAX  DCX
        DB    01011001B    ; INR  DCR   MVI   RRC
        DB    00110101B    ;      LXI   STAX  INX
        DB    01011001B    ; INR  DCR   MVI   RAL
        DB    00010101B    ;      DAD   LDAX  DCX
        DB    01011001B    ; INR  DCR   MVI   RAR
        DB    00111101B    ;      LXI   SHLD  INX
        DB    01011001B    ; INR  DCR   MVI   DAA
        DB    00011101B    ;      DAD   LHLD  DCX
        DB    01011001B    ; INR  DCR   MVI   CMA
```

```
DB    00111101B   ;       LXI   STA   INX
DB    01011001B   ; INR   DCR   MVI   STC
DB    00011101B   ;       DAD   LDA   DCX
DB    01011001B   ; INR   DCR   MVI   CMC
```

Przypisanie instrukcji HLT i kodom nielegalnym długości równej zeru powoduje zaniechanie przepisywania ich do obszaru roboczego Monitora i przekazywanie sterowania bezpośrednio pod odpowiedni adres, uniemożliwiając ich realizację. Fakt ten można wykorzystać przy uruchamianiu programów, jeżeli na danym etapie pragniemy uniknąć wykonania określonej instrukcji. Wystarczy w tym celu chwilowo umieścić w odpowiednim miejscu programu jeden z ww. kodów i ustawić na nim pułapkę. Ten sam efekt można uzyskać umieszczając w programie instrukcję RST, wykorzystywaną do realizacji pułapek.

Poprawne działanie zaprezentowanego algorytmu wymaga nieprzerywalności realizacji całej sekwencji przekazywania sterowania. Rozwiązanie tego zagadnienia jest ściśle związane ze specyfiką architektury konkretnego systemu mikrokomputerowego wykorzystującego mikroprocesor 8080.

MICHAŁ BYSTRIANIN
Centrum Badań Kosmicznych
Warszawa

Zgodnie z zapowiedzią kontynuujemy kącik dla amatorów języka FORTH. Tym razem niezła porcja wiadomości. Ale, co najważniejsze, w oparciu o podany materiał możemy już tworzyć pierwsze programy! Zanim jednak FORTH stanie się przygodą, czeka nas jeszcze trochę żmudnego „przyswajania" podstawowych informacji. Staramy się skrócić do niezbędnego minimum część szkoleniową. Cierpliwości!

FORTH (2)

OPERACJE NA STOSIE

FORTH operuje na stosie danych (PARAMETER STACK) i stosie powrotów (RETURN STACK). Norma języka FORTH79 zawiera 12 słów do bezpośredniej obsługi stosów.

Przy opisie słów przyjęto następującą notację:
stan stosu danych : : =
(<dane na stosie przed wykonaniem słowa>
— <dane na stosie po wykonaniu słowa>)
stan stosu powrotów : : =
(<dane na stosie przed wykonaniem słowa>
— <dane na stosie po wykonaniu słowa>)
dane na stosie : : = $<n_k\ n_{k-1} \ldots n_2\ n_1>$
gdzie: n_1 liczba na szczycie stosu, n_k ostatnia liczba na spodzie stosu (lub ostatnia istotna dla danego słowa)

Powielenie liczby ze szczytu stosu **DUP**
DUP (<n>—<n n>)
Np. 5 DUP .. [CR] 5 5 OK

Poszczególne dyrektywy oddzielane są spacją. Każda kropka powoduje pobranie i wyświetlenie liczby ze szczytu stosu .[CR] to znak końca instrukcji: klawisz "new line")

Powielenie niezerowej liczby ze szczytu stosu **?DUP**
?DUP (<n>— { <n n> — gdy $n \neq 0$ | <n> — gdy $n = 0$ })
Np. 5 ?DUP .. [CR] 5 5 OK
0 ?DUP .. [CR] 0 EMPTY STACK (EMPTY STACK — komunikat o braku danych na stosie)

Zmniejszenie wskaźnika szczytu stosu **DROP**
DROP (<n>—<>)
Np. 3 4 DROP .. [CR] 3 OK

Zamiana dwóch danych ze szczytu stosu **SWAP**
SWAP ($<n_2\ n_1>$—$<n_1\ n_2>$)
Np. 1 5 SWAP .. [CR] 3 OK
(Zauważmy, że 1 5 .. [CR] 5 1 OK)

Skopiowanie na szczyt stosu drugiej od końca liczby na stosie **OVER**
OVER ($<n_2\ n_1>$—$<n_2\ n_1\ n_2>$)
Np. 2 3 OVER...[CR] 2 3 2 OK

Rotacja trzech pozycji ze szczytu stosu **ROT**
ROT ($<n_3\ n_2\ n_1>$—$<n_2\ n_1\ n_3>$)
Np. 3 2 1 ROT...[CR] 3 1 2 OK

Skopiowanie na szczyt stosu wybranej pozycji ze stosu **PICK**
PICK ($<n_k\ n_{k-1} \ldots n_2\ n_1\ i>$ — $<n_k\ n_{k-1} \ldots n_2\ n_1\ n_i>$)
Ostatnia wartość na stosie jest pobierana jako parametr określający położenie na stosie poszukiwanej liczby.
Np. 5 4 8 2 1 3 PICK.....[CR] 8 1 2 8 4 5 OK
Zwróćmy uwagę, że 2 PICK odpowiada OVER.

Rotacyjna zamiana miejscami wybranej liczby elementów ze stosu **ROLL**
ROLL ($<n_k \ldots n_1\ i>$ — $<n_k \ldots n_{i+1}\ n_{i-1} \ldots n_1\ n_i>$)
Ostatnia wartość na stosie jest pobierana jako parametr określający liczbę elementów podlegających rotacji.
Np. 1 2 3 4 5 6 4 ROLL.....[CR] 3 6 5 4 2 1 OK
Zwróćmy uwagę, że: 2 ROLL odpowiada SWAP
3 ROLL odpowiada ROT.

Badanie głębokości stosu **DEPTH**
DEPTH ($<n_k\ n_{k-1} \ldots n_2\ n_1>$ — $<n_k\ n_{k-1} \ldots n_2\ n_1\ k>$)
Wykonanie DEPTH powoduje odłożenie na szczycie stosu liczby odpowiadającej ilości danych na stosie w momencie wywołania słowa.
Np. 1 3 2 4 5 0 DEPTH.[CR] 6 OK

Przeniesienie ostatniej pozycji stosu danych na stos powrotów **> R**
> R (<n>—<>)[<>—<n>]

Przeniesienie ostatniej pozycji stosu powrotów na stos danych **R >**
R > (<>—<n>) [<n>—<>]
Np. 1 3 >R 12 15 R > ..[.. [CR] 3 15 12 1 OK

Skopiowanie ostatniej pozycji stosu powrotów na stos danych **R @**
R @ (<>—<n>) [<n>—<n>]

Np. 1983 > R 11 DUP 12 24 .. R @ ... R @... [CR] 24 12 1983 11 11 1983 OK

OPERACJE ARYTMETYCZNE

Dodawanie +
+ ($<n_2\ n_1>$—$<n_2+n_1>$)
Np. 2 4 +. [CR] 6 OK

Odejmowanie —
— ($<n_2\ n_1>$ — $<n_2 - n_1>$)
Np. 5 7 —. [CR] — 2 OK

Mnożenie *
* ($<n_2\ n_1>$ — $<n_1 * n_2>$)
Np. 2 3 *. [CR] 6 OK

Dzielenie /
/ ($<n_2\ n_1>$ — $<n_2/n_1>$)
Np. 14 7 /. [CR] 2 OK

A teraz kilka bardziej złożonych przykładów. Mamy zdefiniować słowa, które dają nastepujace wyniki:
SLOWO—X (<a b c d> — <((a+c)*b)/(c+d)>)
. SLOWO—X OVER + SWAP ROT >R ROT R> ROT ROT + * SWAP /;
SLOWO—Y (<a b c d e> — <((c + e) * (a + c))/((b + d) * (a + c)/e)>)
: SLOWO—Y > R > R ROT OVER + ROT R> + OVER * R> SWAP OVER / ROT ROT >R ROT R> + * SWAP /;
SLOWO—Z (<a b c> — <(((a + c) * b) + (c + b))/((a + c) * a)>)
: SLOWO—Z OVER OVER + >R ROT DUP >R + R> OVER * R> ROT >R ROT R> * + SWAP /;

Dla „treningu" proponujemy sprawdzić, jaki będzie efekt działania nastepujacego polecenia:
1 2 3 4 + ROT . OVER + . DUP + DUP DUP + .. [CR]

MAREK CZARZYŃSKI
ABAKUS

Tak się niefortunnie składa, że w chwili gdy przygotowujemy lutowe wydanie mikroKLANU, INFORMATYKA dociera do Czytelników z pewnym opóźnieniem. Być może dzięki naszym staraniom numer drugi faktycznie ukaże się w lutym, numer trzeci w marcu itd., zgodnie z kalendarzem. Na wszelki wypadek podajemy już teraz program obliczający datę Wielkanocy.

W połączeniu z algorytmami „kalendarzowymi" (INFORMATYKA nr 9/83) można wzbogacić mikrokomputer w kalendarz, co pozwoli na prowadzenie harmonogramu zajęć, spotkań, spraw do załatwienia, terminów etc.

Procedura „Wielkanoc"

```
10 REM PROGRAM PODAJE DATE WIELKANOCY WG KALENDARZA
20 REM GREGORIANSKIEGO CZYLI OD ROKU 1583 WŁACZNIE.
30 REM DLA DAT WCZESNIEJSZYCH (KALENDARZ JULIANSKI)
40 REM NALEŻY PODSTAWIC:
50 REM M = 15 ORAZ P = 6 (PATRZ LINIE 130 oraz 150).
60 PRINT "ROK?";
70 INPUT Y
80 LET A = Y—INT(Y/19)*19
90 LET B = Y—INT(Y/4)*4
100 LET C = Y—INT(Y/7)*7
110 LET T = INT(Y/100)
120 LET P = 4+T—INT(T/4)
130 LET M = P—INT(P/7)*7
140 LET P = 15+T—INT(T/4)—INT((8*T+13)/25)
150 LET P = P—INT(P/30)*30
160 LET P = 19*A+P
170 LET D = P—INT(P/30)*30
180 IF D < 28 THEN 220
190 IF D = 29 THEN 210
200 IF A < = 10 THEN 220
210 LET D = D—1
220 LET P = 2*B+4*C—D+M
230 LET P = P—INT(P/7)*7
240 LET D =D+P
250 IF D > 9 THEN 290
260 LET D = D+22
270 PRINT "WIELKANOC: " ;D; "MARCA"; Y
280 STOP
290 LET D = D—9
300 PRINT "WIELKANOC:";D;"KWIETNIA"; Y
310 END
```

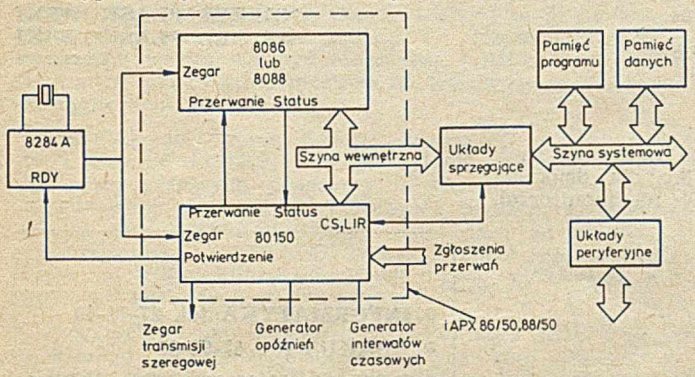
Przykłady pracy programu:

```
ROK? 1900
WIELKANOC: 15 KWIETNIA 1900
ROK? 1921
WIELKANOC: 27 MARCA 1921
ROK? 1984
WIELKANOC: 22 KWIETNIA 1984
ROK? 2000
WIELKANOC: 23 KWIETNIA 2000
ROK? 4356
WIELKANOC: 1 KWIETNIA 4356
```

Opracował
JAKUB TATARKIEWICZ
na podstawie „PPC Calculator Journal", V1ON8P4.

Krzemowy CP/M

Popularny system operacyjny CP/M przechowywany jest zazwyczaj na dyskietce. Niedawno firma INTEL wprowadziła na rynek specjalny układ VLSI 80150, który realizuje funkcje CP/M 86 (udoskonalona wersja CP/M 80 dla systemów iAPX86, iAPX88 oraz iAPX186). Przeznaczony jest on do zastosowania w mikrokomputerach i umożliwia korzystanie z systemu operacyjnego nawet w urządzeniach nie wyposażonych w dyski elastyczne.



Schemat blokowy systemu mikroprocesorowego wykorzystującego układ 80150 przedstawiono na rysunku. Zastosowanie 80150 jest najbardziej ekonomiczną formą realizacji CP/M 86 w mikrokomputerze. Ponieważ wszystkie funkcje sprzętowe wymagane przez system operacyjny zostały zrealizowane w jednym układzie scalonym, oszczędza się na kosztach związanych z projektowaniem mikrokomputera. Zmniejszają się koszty montażu i testowania urządzeń. Wzrasta równocześnie niezawodność systemu, gdyż programy — w przypadku błędu — nie mogą zniszczyć oprogramowania systemowego zawartego w pamięci RAM.

Jeżeli system nie jest wyposażony w stację dysków elastycznych, można ustalić odpowiedni blok pamięci RAM, który będzie traktowany przez system operacyjny jak dyskietka. Realizowanie takich funkcji, jak otwieranie i zamykanie zbiorów, przechowywanie programów itp. — odbywa się analogicznie jak przy stosowaniu dyskietek; z tą tylko różnicą, że wzrasta efektywność systemu, jako że wszystkie operacje wykonywane są w RAM. Programy użytkowe mogą być ładowane przy użyciu konwencjonalnych technik sprzęgających (z pamięci kasetowej, domenowej, stałej) lub nawet z klawiatury.

Opracował AJP
na podstawie
„SOLUTIONS" I/II 1983

- Istnieje możliwość uzyskania (w drodze wymiany) programów dla ZX SPECTRUM. Piotr Parlewicz (tel. w Warszawie: 42-40-94) posiada już zbiór ponad 80 gier i programów użytkowych. Wśród tych ostatnich warto wymienić: **PASCAL, FORTH, ASSEMBLER/DISASSEMBLER/EDITOR, Slow Loader** (ładowanie programów z ZX81), **VU-File** (baza danych).
- Wydany kilka miesięcy temu w niewielkim nakładzie, ale za to na kredowym papierze, **„Katalog Skrócony CEMI 82/83"** (cena 400 zł) jest już niestety dostępny tylko na czarnym rynku. Za ten skrót ze skrótów niektórzy płacą 3 tys. zł!
- BOINTE przy Przemysłowym Instytucie Elektroniki (Warszawa, Marynarska 10, tel. 47-07-62) wydaje zeszyty **„Systemów Mikroprocesorowych"**, w cenie 345 zł za egzemplarz (wyjątkowo drogo!). Można w nich znaleźć opisy profesjonalnych urządzeń lub rozwiązań wykorzystujących układy mikroprocesorowe. Po dłuższej przerwie ukazały się ostatnio zeszyty 11, 12 i 13.
- Przypominamy. Trwa **konkurs na program gry edukacyjnej** organizowany przez klub ABAKUS. Do wygrania m.in. ZX SPECTRUM 48 K (cena „wolno-rynkowa" — ok. 160 tys. zł).
- **Klub Użytkowników Mikroprocesorów (KUM)** zajmuje się wymianą informacji i usług. Ankietę klubową można otrzymać od założyciela klubu — Andrzeja Drożniaka; ul. Pytlasińskiego 14 m 4, 00-707 Warszawa; tel. dom. 41-26-01.

U nas to kawał żmudnej i niewdzięcznej roboty... W krajach rozwiniętych — projektowanie precyzyjnych detali mechanicznych, płytek drukowanych, struktur układów scalonych — to czynności wykonywane przy użyciu mikrokomputerów. Rzecz nie tylko w przeprowadzeniu obliczeń — efektem końcowym jest precyzyjny rysunek (niezależnie od innych nośników informacji wykorzystywanych potem do sterowania urządzeniami technologicznymi).

Do niedawna dla utrwalenia wyników pracy na papierze stosowany był ploter. Od kilku lat ogromnie popularne stało się wykorzystywanie do tych celów drukarek mozaikowych, które oprócz liter mogą stawiać punkty układające się w obraz. Na Targach Poznańskich zaprezentowano drukarkę D-200 (polska!), którą można wykorzystać do celów graficznych. Lecz jak to często bywa z prezentowanymi tam wyrobami — drukarka taka nie jest sprzedawana na krajowym rynku. Dlatego też poniższy przykład opracowany został na komputerze BBC (patrz rozmowa w poprzednim wydaniu mikroKLANU) współpracującym z drukarką SEIKOSHA. Warto zwrócić uwagę, że banalne w sumie oprogramowanie w połączeniu z wcale nie tak znów wyrafinowanym sprzętem — daje rewelacyjne wyniki! No cóż, wydawałoby się, że możemy tylko pomarzyć... Tymczasem grupa zapaleńców już postanowiła przerobić poczciwą DZM 180, tak aby uzyskać wyniki nie gorsze od prezentowanych w poniższym tekście. W jednym z najbliższych wydań mikroKLANU opublikujemy ich doświadczenia.

Kopiowanie ekranu na drukarkę graficzną

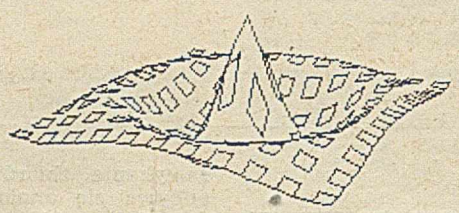
Jeżeli nie dysponujemy ploterem można napisać program kopiujący obraz z monitora CRT na drukarkę mozaikową. W oprogramowaniu mikrokomputera BBC — użytego w przedstawionym rozwiązaniu — znajduje się funkcja POINT (X, Y), która analizuje punkt o współrzędnych X, Y na ekranie monitora. Przyjmuje ona wartość odpowiadającą kolorowi danego punktu.

Jeżeli — na przykład — wykorzystywane są tylko dwa kolory, funkcja POINT przyjmuje wartość 0 dla punktu czarnego, a dla punktu białego — wartość 1.

Maksymalna rozdzielczość monitora ekranowego wynosi 1280 × 1024 punkty. W zależności od trybu pracy, rozdzielczość monitora można zmniejszyć (na plamę świetlną składa się kilka lub kilkanaście punktów):
tryb 0 — 640 × 256
tryby 1, 4 — 320 × 256
tryby 2, 5 — 160 × 256
tryby 3, 6, 7 — nie dają możliwości graficznych.

W trybie 4, dla którego opracowano poniższy przykład, punkt świetlny ma rozmiar 4 × 4, co należało uwzględnić przy pisaniu programu.

W drukarce znakowo-mozaikowej SEIKOSHA, za pomocą której kopiowano obraz z monitora, znaki tworzone są z matrycy punktów o wymiarach 6 × 7 (6 kolumn po 7 punktów każda). Drukowanie graficzne jest realizowane w sposób następujący. Do drukarki należy przesłać kod drukowania graficznego VDU 1,8, a następnie ciąg liczb. Każda liczba jest kodem jednej kolumny zawierającej punkty odbijane przez siedem igieł. Liczba ta musi być zawarta w przedziale od 128 do 128 + 2^7 (liczby od 0 do 127 zarezerwowane są dla kodów ASCII).

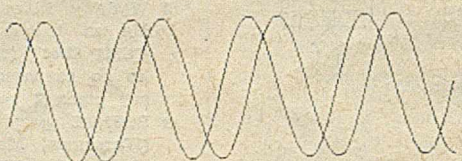


Rys. 1. Obraz graficzny funkcji Z = cos (a/1,5) · EXP (—a/5) · 5, gdzie $a = \sqrt{x \cdot x + y \cdot y}$ – zrealizowany na drukarce mozaikowej

Oto przykład określania kodu:

```
o
•
•
o A = 128+(2¹+2²+2⁴+2⁶)
•
o
•
```

VDU 1 oznacza, że następny znak przeznaczony jest tylko dla drukarki (nie jest wyświetlany na monitorze).



Rys. 2. Wykresy funkcji y = sin x i y = = cos x — zrealizowane na drukarce mozaikowej

Jeżeli zmiennym A, B, C zostaną przypisane żądane kody, zapis instrukcji drukowania może wyglądać następująco:
VDU 1, 8, A, B, C,...
lub też
VDU 1, 8
VDU 1, A
VDU 1, B
VDU 1, C

Włączenie drukarki następuje przez VDU 2, a wyłączenie drukarki przez VDU 3.

Dla kodów ASCII drukarka drukuje do 80 znaków w rzędzie, tak więc maksymalna rozdzielczość w rzędzie wynosi 480 punktów (80 × 6).

Procedura kopiowania została napisana w języku BASIC:

```
10 DEFPROCHARTCOPY
20 VDU2, 1,8:FORY% = 812T00STEP—28:
                                VDU13
30 FORX% = 0T01200STEP4:A% = 128
40   FORN% = 0T027STEP4
50     IF POINT(X%,   Y%—N%) = 1   GO-
                                SUB90
60     NEXT : VDU1,A% : NEXT
70 NEXT. : VDZ13,3,4,1,15
80 ENDPROC
90 ONN%DIV4 + 1GOTO100,110,120,130,140,150,
                                160
100 A% = A%+1:RETURN
110 A% = A%+2:RETURN
120 A% = A%+4:RETURN
130 A% = A%+8:RETURN
140 A% = A%+16:RETURN
150 A% = A%+32:RETURN
160 A% = A%+64:RETURN
```

Za pomocą powyższego programu skopiowano wykresy funkcji przedstawionych na rysunkach.

Czas kopiowania monitora
— przy całym obrazie z monitora ok. 15 min
— przy jednym wykresie ok. 4 min
— przy kilku (rzędu 10) ok. 6 min.

Istnieje również możliwość drukowania negatywu obrazu z monitora. Należy wówczas zmienić w wierszu 50 POINT (X%, Y%—N%) = 1 na POINT (X%, Y%—N%) = 0.

ANDRZEJ PLĄSKOWSKI
WITOLD PLĄSKOWSKI
Warszawa

- **INFORMATYKA tel. 27-71-40**
- **ABAKUS tel. 42-91-85**
- **KUM tel. 41-26-01**
- ?

— prowadzi Andrzej J. Piotrowski (tel. dom. 48-22-85)